School Management Software Documentation

Table of Contents

Module 3: Introduction to Spring boot and Spring Framework

3. Student Management Microservice

GIT URL-

3.1. Fetch All Students

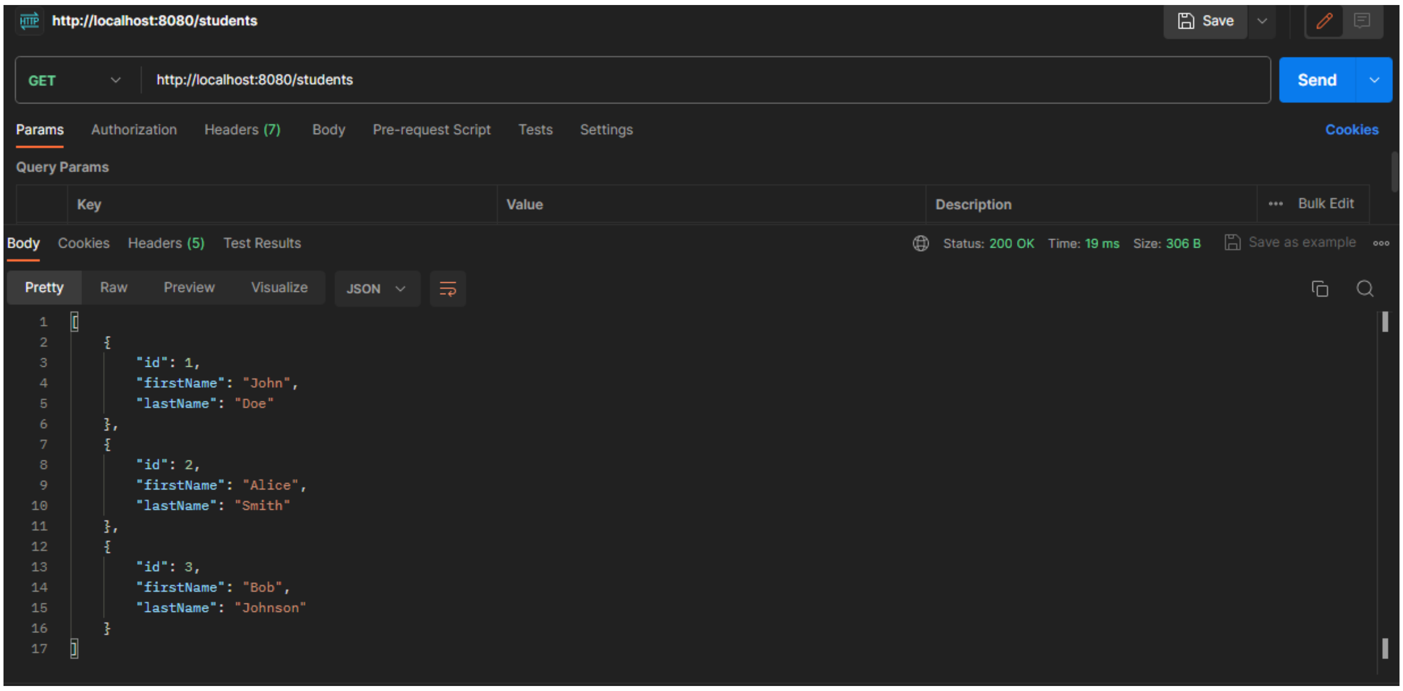
Implementing an API to fetch all students from the database.

Endpoint: GET /students

http://localhost:8080/students

Description: This API retrieves a list of all students from the database.

Screenshot:



GET http://localhost:8080/students

Status: 200 OK Time: 19 ms Size: 306 B

```
[
    {
        "id": 1,
        "firstName": "John",
        "lastName": "Doe"
    },
    {
        "id": 2,
        "firstName": "Alice",
        "lastName": "Smith"
    },
    {
        "id": 3,
        "firstName": "Bob",
        "lastName": "Johnson"
    }
]
```

3.2. Fetch Single Student

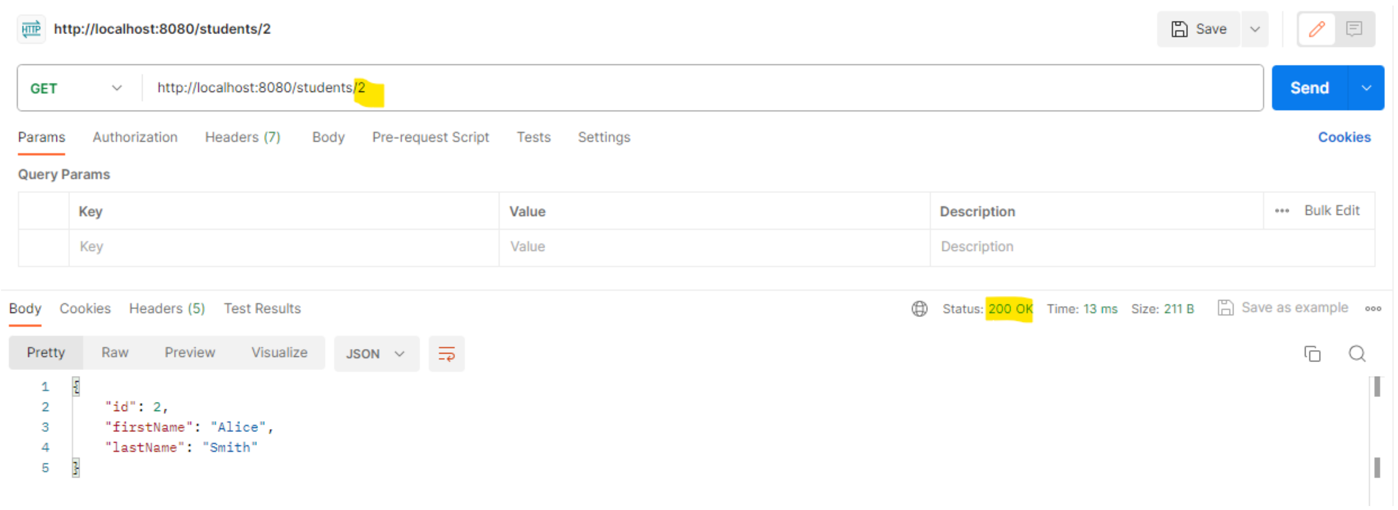
Implementing an API to fetch a single student by ID from the database.

Endpoint: GET /students/{id}

http://localhost:8080/students/2

Description: This API retrieves a single student based on the provided ID.

Screenshot:



http://localhost:8080/students/2

GET http://localhost:8080/students/2

Params | Authorization | Headers (7) | Body | Pre-request Script | Tests | Settings

Query Params

Key	Value	Description
Key	Value	Description

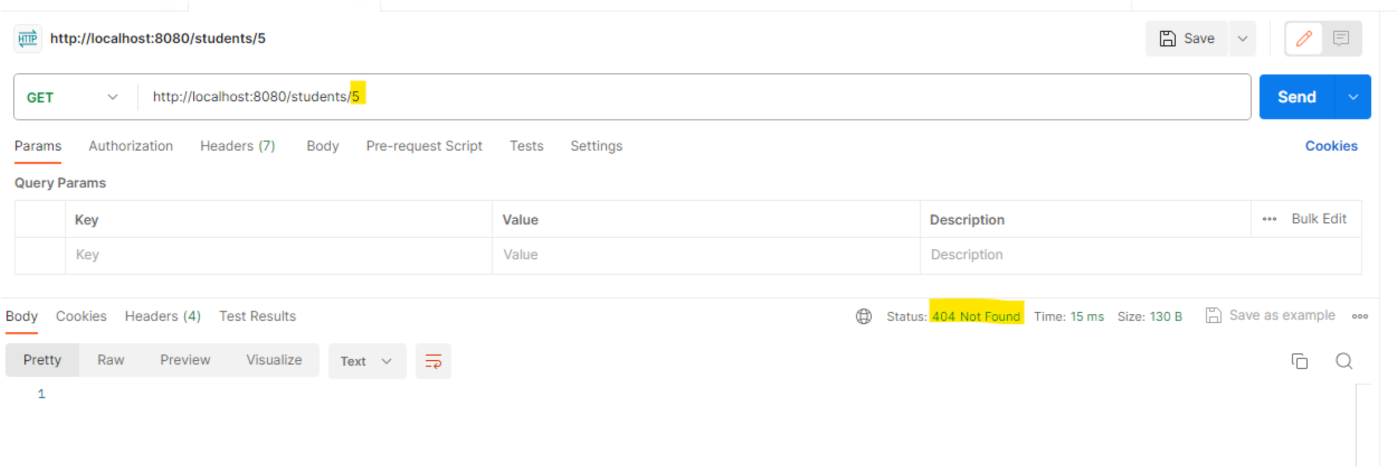
Body | Cookies | Headers (5) | Test Results — Status: 200 OK Time: 13 ms Size: 211 B

Pretty | Raw | Preview | Visualize | JSON

```
{
    "id": 2,
    "firstName": "Alice",
    "lastName": "Smith"
}
```

Trying to retrieve invalid student for example

http://localhost:8080/students/5   we get **404 Not Found**



http://localhost:8080/students/5

GET http://localhost:8080/students/5

Params | Authorization | Headers (7) | Body | Pre-request Script | Tests | Settings

Query Params

Key	Value	Description
Key	Value	Description

Body | Cookies | Headers (4) | Test Results — Status: 404 Not Found Time: 15 ms Size: 130 B

Pretty | Raw | Preview | Visualize | Text

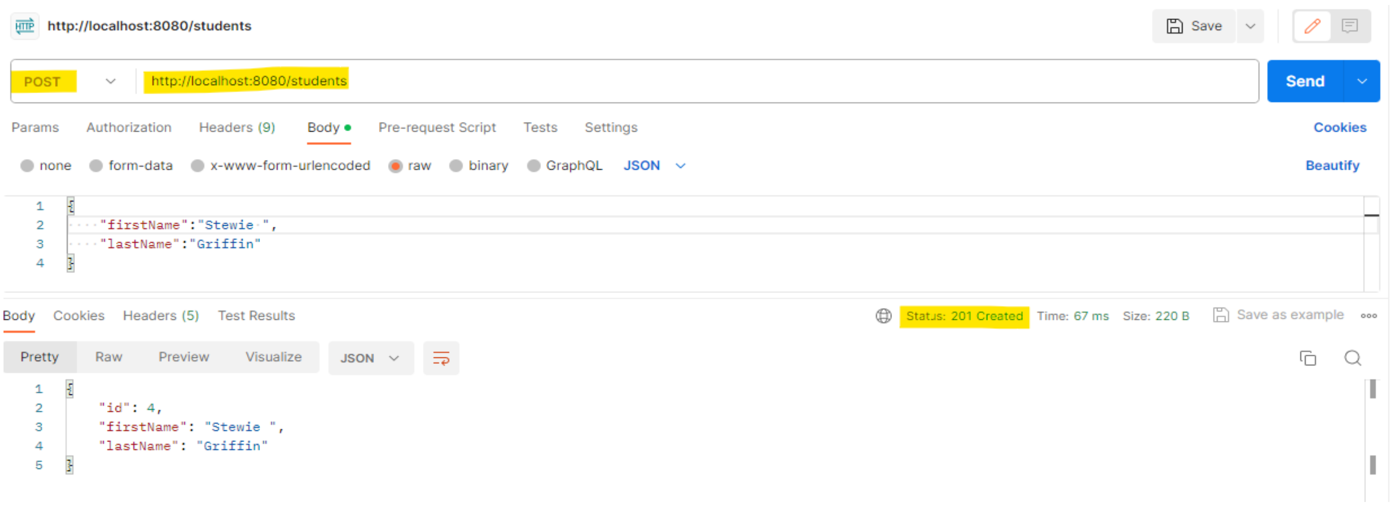
3.3. Create New Student

Implementing an API to create a new student and store it in the database.

Endpoint: POST /students

http://localhost:8080/students

Description: This API creates a new student with the provided information.
Screenshot:



http://localhost:8080/students

POST http://localhost:8080/students

Params | Authorization | Headers (9) | Body | Pre-request Script | Tests | Settings

none | form-data | x-www-form-urlencoded | raw | binary | GraphQL | JSON

```
{
    "firstName":"Stewie ",
    "lastName":"Griffin"
}
```

Body | Cookies | Headers (5) | Test Results — Status: 201 Created Time: 67 ms Size: 220 B

Pretty | Raw | Preview | Visualize | JSON

```
{
    "id": 4,
    "firstName": "Stewie ",
    "lastName": "Griffin"
}
```

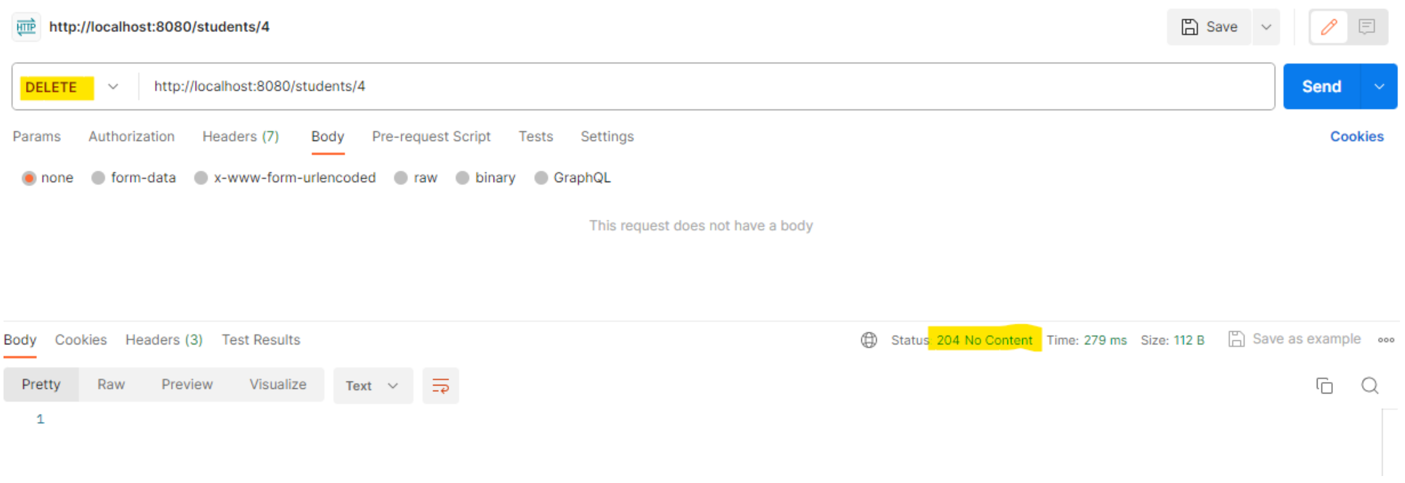
3.4. Delete Student

Implementing an API to delete a student by ID from the database.

Endpoint: DELETE /students/{id}

Description: This API deletes a student based on the provided ID.

Screenshot:



http://localhost:8080/students/4

DELETE http://localhost:8080/students/4

Body: none

This request does not have a body

Status 204 No Content Time: 279 ms Size: 112 B

3.5. Update Student

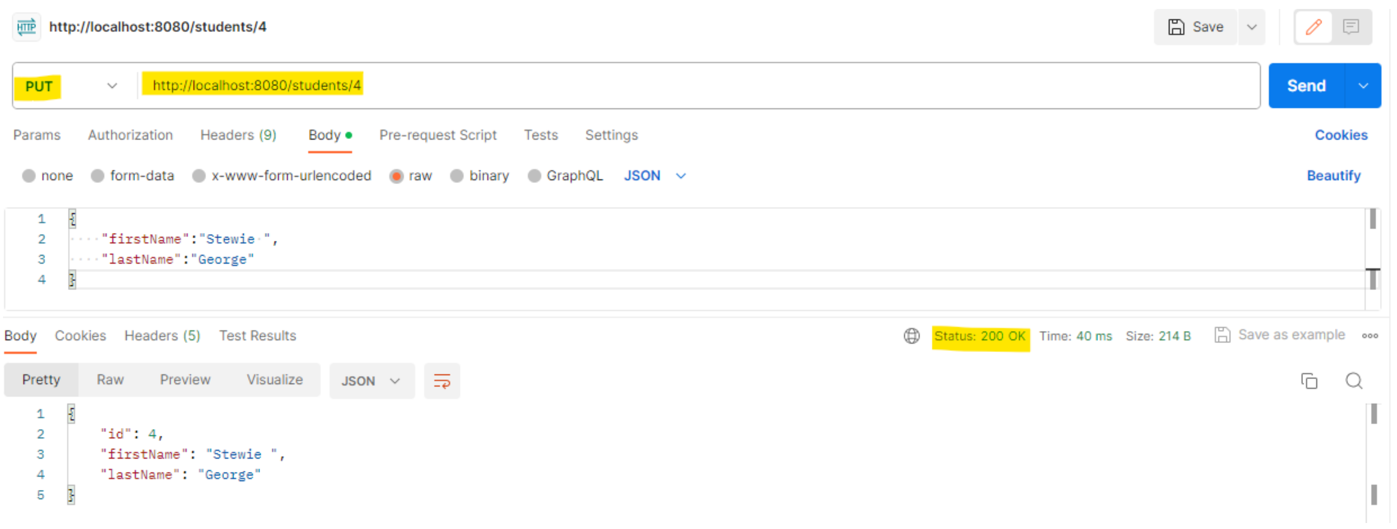
Implementing an API to update a student's information in the database.

Endpoint: PUT /students/{id}

http://localhost:8080/students/4

Description: This API updates a student's information based on the provided ID.

Screenshot:



http://localhost:8080/students/4

PUT http://localhost:8080/students/4

```
{
    "firstName":"Stewie ",
    "lastName":"George"
}
```

Status: 200 OK Time: 40 ms Size: 214 B

```
{
    "id": 4,
    "firstName": "Stewie ",
    "lastName": "George"
}
```